Санкт-Петербургский государственный университет

КИРИЛЛОВ Денис Андреевич

Выпускная квалификационная работа

Применение технологий распределенных реестров в системах электронного голосования

Уровень образования: магистратура
Направление: 02.04.02 «Фундаментальная информатика и информационные технологии»
Основная образовательная программа: ВМ.5502 «Вычислительные технологии»

Научный руководитель:
доцент кафедры КММС,
кандидат физ.-мат. наук,
Корхов Владимир Владиславович

Рецензент:
ведущий программист ООО «ОСЕНСУС АРМ»,
кандидат физ.-мат. наук,
Балян Сероб Гургенович

Санкт-Петербург
2020

Содержание

Введение

Несмотря на то, что технологии развиваются стремительными шагами (уже стало обыденностью не использовать наличные деньги и не иметь нигде дела с бумажными носителями), во многих странах для разного рода голосований (в том числе и выборов) до сих пор используются бумажные бюллетени. Это связано с необходимостью обеспечения должного уровня надежности системы голосования, так как ставки в этих вопросах могут быть очень высоки. В то же время попытки внедрить существующие протоколы электронного голосования в некоторых странах (Эстония [1], Австралия[2]) показали, что не все проблемы безопасности были решены [3,4], тем самым увеличив скептицизм государственных органов по отношению к новым решениям.

Перевод голосования в цифровую среду мог бы помочь увеличить прозрачность данной процедуры для конечных пользователей и повысить безопасность (при условии, что выбрана правильная модель). Также это могло бы существенно уменьшить затраты на проведение данной процедуры и снять нагрузку с проверяющих.

Тем не менее, голосование (особенно выборы) это такая процедура, в которой никто никому не доверяет и необходимо отталкиваться от мысли, что систему попытаются сломать (подтасовать результаты, сорвать сам процесс и т.д.) не только внешние злоумышленники, но и сами участники системы, как голосующие, так и организаторы. В таких условиях традиционные подходы, основанные исключительно на криптографических техниках, не всегда могут давать приемлемый результат. Требуются некоторые дополнительные инструменты, которыми могут стать технологии распределенных реестров.

Постановка задачи

Требуется исследовать протоколы электронного голосования (которые могут быть практически реализованы), а также усовершенствовать один из этих протоколов, путем использования технологии распределенных реестров, для того чтобы повысить прозрачность и доверие к системе голосования. Помимо этого необходимо провести тесты на производительность, оценить полученные результаты и, если они будут неудовлетворительными, предложить пути исправления и усовершенствования.

Обзор литературы

Проблема электронного голосования является достаточно сложной, из-за большого количества требований, предъявляемых к готовой системе. Поэтому существует достаточно много работ посвященных этой теме, которые предлагают разные подходы, в той или иной степени покрывающие необходимые ограничения. Также используются совершенно разные криптографические техники: слепая подпись, кольцевая подпись, доказательство с нулевым разглашением, гомоморфное шифрование, криптосистема Пейе [5]. В этом разделе для краткости CA будет обозначать орган, который организует процедуру голосования.

В статье авторов Hardwick и Naeem [6] предлагается использовать слепую подпись от CA над публичным ключом и цифровым подтверждением (digital commitment) голоса, чтобы получить право голосовать. После этого пользователь отправляет всем узлам сети свой бюллетень (публичный ключ, цифровое подтверждение, подпись от CA). Затем узлы проверяют подпись и записывают данный голос в реестр, а избиратель получает id, под которым его голос записан. Если он хочет изменить свой выбор, то он формирует новый бюллетень, содержащий id голоса, который надо заменить, и данные аналогичные первому бюллетеню. После окончания фазы голосования, избиратель отправляет данные для открытия цифрового подтверждения своего голоса, id голоса и подпись CA на них. Происходит подсчет голосов, результаты которого затем записываются в реестр.

Так как используется технология блокчейн, то каждый избиратель на любой стадии может проверить, что его голос присутствует в сети и будет учтен. Определенная степень сокрытия голоса достигается использованием подписанных слепой подписью публичный ключей. Голос будет учтен не более одного раза, так как в каждый момент времени в реестре присутствует

только один не замененный другим голос, принадлежащий одному публичному ключу.

Немного другой метод предложен в статье [7]. Голосующие должны иметь ID и PIN, с помощью которых можно войти в систему, в которой создается “кошелек” для каждого пользователя (возможно использование доказательств с нулевым разглашением), через который он будет взаимодействовать со смарт-контрактом, который и будет записывать все голоса в сеть блокчейн. Избиратель отдает свой голос путем вызова функции смарт контрактов, соответствующих избирательному округу, передавая в качестве аргумента номер кандидата и получая id транзакции, в которой и содержится голос, в качестве результата. Смарт-контракты на всем протяжении выборов хранят в себе текущий прогресс (то есть есть переменная, в которой лежит количество голосов за того или иного кандидата).

Для того чтобы достичь анонимности используется специальная структура транзакций, в которой отсутствует поле “отправитель”. Избиратель не может изменить свой голос. Проверить что голос был учтен можно лишь придя в СА и раскрыв свою личность. К тому же в самой статье говорится, что нет гарантии, что голос будет посчитан и притом корректно.

В статье авторов Yavuz и Koç [8] система электронного голосования реализована на платформе Ethereum, где организатором выборов создается смарт контракт, в который добавляются адреса кошельков тех людей, кто допущен к голосованию. После этого избиратели вызывают функцию данного смарт-контракта, передавая в качестве параметра номер кандидата (изменить выбор нельзя). В конце выборов организатором вызывается функция, которая возвращает номер кандидата, который набрал больше всего голосов.

Такой подход имеет ряд недостатков. Например, отсутствует анонимность. Это происходит из-за того, что авторами не предполагается никаких криптографических техник, а структура сети блокчейн Ethereum прозрачна и любой участник видит содержимое транзакций. Помимо этого избирателю надо платить комиссию за каждый вызов смарт-контракта. Также предложенный подход плохо масштабируется из-за небольшого количества обрабатываемых транзакций в минуту.

Yi Liu и Qi Wang [9] предлагают ввести определенное количество инспекторов, которые будут принимать участие в фазе голосования. Изначально избиратель генерирует пару ключей, один из которых (публичный) отправляет в CA вместе с данными для аутентификации. CA проверяет избирателя и публикует список зарегистрированных публичный ключей, владельцы которых допущены к голосованию. Затем избиратели формируют свои голоса и отправляют их на слепую подпись к CA с помощью транзакции, в которой в качестве отправителя указывают свой публичный ключ. CA проверяет, что данный избиратель имеет право голосовать и еще не отдал свой голос, и возвращает подписанные данные. Затем избиратель отправляет то же самое на подпись всем инспекторам. После успешного прохождения этой стадии у него оказывается его голос, подписанный всеми контролирующими участниками. Голосующий генерирует новую пару ключей и отправляет подписанный голос в CA, в качестве отправителя указав новый публичный ключ. После окончания голосования CA публикует результаты.

В статье [10] предложен подход, который в качестве криптографических техник использует криптосистему Пейе (Paillier) [5] и короткую связывающую кольцевую подпись (SLRS) [11]. Сам протокол голосования выглядит следующим образом: сначала инициализируется смарт-контракт, затем в блокчейн загружаются параметры для криптосистемы Пейе и

кольцевой подписи. После этого пользователь с использованием данных от администратора входит в систему и скачивает параметры кольцевой подписи и публичный ключ криптосистемы Пейе, генерирует свои ключи с помощью SLRS и отправляет публичный ключ смарт-контракту, тем самым завершая свою регистрацию в системе. Во время фазы голосования избиратель формирует свой голос, шифрует его с помощью криптосистемы и вызывает функцию, которая предоставляет данные, которые доказывают, что зашифрованный голос действительно зашифровал номер одного из существующих кандидатов и отправляет эти данные вместе с голосом смарт-контракту, который после валидации добавляет к голосу зашифрованный ноль, подписывает данные и возвращает их избирателю. Тот проверяет подпись и накладывает на полученное сообщение свою кольцевую подпись, затем снова отправляет его смарт-контракту. Смарт-контракт проверяет, что избиратель еще не голосовал и записывает голос в блокчейн. Когда наступает фаза подсчета, смарт-контракт подписывает сумму зашифрованных голосов и отправляет ее администратору. Тот с помощью приватного ключа криптосистемы расшифровывает сумму и формирует данные, с помощью которых можно проверить, что сумма, полученная им, действительно является подлинной и отправляет эти данные обратно смарт-контракту, который вычисляет количество голосов, набранных каждым кандидатом, и записывает эти результаты в блокчейн.

Недостаток данного подхода в том, что если собрать приватные ключи всех пользователей из одного набора (набор зашифрованных нулей, которые добавляются к зашифрованному голосу избирателя, чтобы внести некоторую случайность), можно узнать как проголосовали эти участники. В качестве решения можно увеличивать размер такого набора, чтобы повысить сложность такой реконструкции для злоумышленника.

Большинство существующих решений, базируются на использовании известных платформ распределенных реестров, однако существует также и большое количество работ, где авторы, воодушевившись основными преимуществами, которые предоставляет блокчейн, создают продукты без использования существующих платформ. В статье [12] описывается принцип работы созданной системы электронного голосования без использования готовых блокчейн платформ. Реализуемое решение построено на связке NodeJS-ReactJS, распределенность достигается путем развертывания различных узлов на базе Heroku, взаимодействующими при помощи PubNub - реализации издатель-подписчик. В статье описан пример развертывания системы для случая территориально распределенных голосований, когда на каждый участок развертывается по отдельному узлу блокчейн сети. Описанная система представляет из себя традиционную реализацию публичного блокчейна, поддержание которого обеспечивается за счет поддержки возможности майнинга, то есть используются основные принципы алгоритма PoW (доказательство работы). В основе решаемой математической PoW задачи также лежит принцип односторонних функций, а алгоритм корректировки сложности задачи схож с принципом функционирования Bitcoin. Одной из целей исследования авторы ставили сокращение объема хранимых данных. Как следствие, в системе используется криптография на эллиптических кривых (ECC), обеспечивающая меньший размер ключа, сокращая тем самым объем хранимых данных по сравнению с RSA-системами.

Пример реализации еще одной системы электронного голосования без использования платформ распределенных реестров описан в статье [13]. Кроме того, здесь также используется ECC. Система реализована под OS Ubuntu с использованием языка программирования Python. Как и большинство систем, система призвана обеспечить удовлетворение основных

требований к электронному голосованию, включая поддержку возможности переголосования пока время окончания процедуры не достигнуто. С точки зрения архитектуры реализованная система является модульной системой и состоит из трёх независимых компонент:

1. Синхронизированная модель записей голосования на основе технологий распределенных реестров, чтобы избежать подделки голосов;
2. Модель учетных данных пользователя, основанная на криптографии на основе эллиптических кривых (ECC) для обеспечения аутентификации и невозможности отказа от авторства;
3. Модель отзыва, которая позволяет избирателям изменить свой голос до установленного срока.

Для большинства блокчейн приложений сохраняется иерархический принцип ведения распределенного журнала: журнал состоит из блоков, блоки включают в себя транзакции. Применительно к большинству голосований это означает, что каждый голос является транзакцией, которые впоследствии будут объединены в блоки на основании размера блока или же таймаута и сохранены в общий журнал. В случае схемы, описанной в данной работе, блок состоит исключительно из одного бюллетеня от одного пользователя, то есть отсутствует дополнительный уровень блокчейн иерархии. Подобная архитектура свойственна для Byteball Bytes DAG DLT [14], где понятие блока отсутствует в принципе и каждая транзакция содержит информацию о предыдущей. Система позволяет проводить исключительно публичные голосования без сокрытия каких-либо деталей проводимой процедуры.

Авторы статьи [15] обратили внимание на иной аспект современных протоколов электронного голосования - PKI (инфраструктура открытых ключей). Большинство существующих систем блокчейн-голосования основаны на связке криптографических ключей, надежность которых

гарантируется теорией чисел для современных компьютеров. Тем не менее стремительное развитие квантовых технологий является угрозой для современных криптографических алгоритмов, что может стать серьезной угрозой для существующих систем, использующих криптографические подходы. В данной статье уделяется особое внимание вопросу анонимизации голосующих. Предложенная в статье схема комбинирует подходы кольцевой подписи и анонимизации криптосистемы Нидеррайтера. Защита от квантовых атак, ставшая основной особенностью описанной системы, обеспечивается за счет использования модифицированной криптосистемы Нидеррайтера с целью распределения частичных приватных ключей для каждого избирателя. Алгоритм основан на коде, и его безопасность сводится к проблеме декодирования синдрома из теории кодирования. Это NP-полная проблема, которую трудно решить даже перед квантовыми компьютерами с высокой вычислительной мощностью. В статье приведено формальное доказательство безопасности алгоритма. Предложенная в статье схема характеризуется обратной зависимостью безопасности и эффективности системы. Оптимальное значение для безопасности и производительности может быть обеспечено только для небольших голосований. Увеличение количества голосующих приводит к повышению уровня безопасности системы, оказывая сильное влияние на значение производительности системы.

В большей степени вопросы производительности были затронуты в [16]. В статье рассматриваются различные конфигурации блокчейн платформы для проведения голосований с целью сравнительного анализа с точки зрения производительности и масштабируемости систем. Реализуемость подобного анализа становится возможной в связи с большой гибкостью используемой блокчейн платформы Multichain, позволяющей

управлять не только традиционными для подобных систем параметрами вроде размера блока или транзакции, но и уровнем доступа к блокчейн сети.

Глава 1. Моделирование предметной области

1.1. Концептуальная модель

Так как наша конечная цель - это построение информационной системы, которая может иметь реальное применение (например в государственных выборах), то модель, лежащая в ее основе, должна отражать законы, которые регулируют данную процедуру. В России есть законы о голосовании (в том числе и электронном), например:

1. Федеральный закон от 12.06.2002 N 67-ФЗ (ред. от 01.04.2020) "Об основных гарантиях избирательных прав и права на участие в референдуме граждан Российской Федерации";
2. Федеральный закон от 29.05.2019 N 103-ФЗ "О проведении эксперимента по организации и осуществлению дистанционного электронного голосования на выборах депутатов Московской городской Думы седьмого созыва";
3. Федеральный закон "О Государственной автоматизированной системе Российской Федерации "Выборы" от 10.01.2003 N 20-ФЗ (последняя редакция).

Но во втором законе нет конкретного порядка проведения, а в третьем описан порядок использования терминалов, которые ставятся на избирательных участках, поэтому будем строить модель на основе ФЗ N 67 ст. 64 "Порядок голосования".

Так как закон предполагает либо голосование с использованием бумажных бюллетеней, либо через систему ГАС “Выборы”, то нет смысла

учитывать в модели такие требования к порядку проведения процедуры как “Заполненные бюллетени опускаются избирателями, участниками референдума в опечатанные (опломбированные) ящики для голосования либо в технические средства подсчета голосов при их использовании.”, которые относятся исключительно к традиционному голосованию. Поэтому здесь мы упростим модель, выделив основные свойства, которые предъявляются к процедуре голосования, а следовательно и к информационной системе, которая ее реализует:

1. Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей;
2. Каждый голосует лично, не допускается голосование за кого-то другого;
3. Можно проголосовать только один раз;
4. Нельзя узнать как проголосовал кто-то другой;
5. Нельзя узнать промежуточные результаты;
6. После голосования у избирателя не должно оставаться данных, с помощью которых он может доказать третьим лицам как именно он проголосовал.

Здесь стоит отметить разницу между 4 и 6 пунктом. Четвертый пункт подразумевает, что нельзя узнать результаты кого-то другого, если он сам их не раскроет, а шестой, что кто-то другой, даже если бы и хотел, то не смог бы раскрыть свой голос.

1.2. Расширенная модель

Свойства, которым соответствует традиционное голосование, могут быть дополнены еще одним, которое сложно выполнить при обычных выборах, но которое может быть выполнено, если используются информационные технологии.

Это свойство следующее: “Избиратель может проверить, что его голос был учтен”. Такое требование кажется взаимоисключающим с требованием отсутствия любых данных после голосования, которые могут подтвердить выбор голосующего. Однако в некоторых работах, описывающих протоколы электронного голосования [10], предлагают такого достичь с помощью криптосистемы Пейе [5] и кольцевых подписей [17]. Таким образом свойства, которым должна удовлетворять дополненная модель, лежащая в основе информационной системы электронного голосования, следующие:

1. Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей;
2. Каждый голосует лично, не допускается голосование за кого-то другого;
3. Можно проголосовать только один раз;
4. Нельзя узнать как проголосовал кто-то другой;
5. Нельзя узнать промежуточные результаты;
6. После голосования у избирателя не должно оставаться данных, с помощью которых он может доказать третьим лицам как именно он проголосовал;
7. Избиратель может проверить, что его голос был учтен.

1.3. Упрощение

1.3.1. Базовое упрощение

Попробуем сократить число “жизненно необходимых” свойств так, чтобы конечная система стала проще, но не сильно отличалась от той процедуры, которая сейчас существует в законе.

Рассмотрим “отсутствие данных, для доказательства своего выбора третьим лицам”. На самом деле данное свойство в традиционной системе голосования можно достаточно просто обойти. Например, можно

сфотографировать свой заполненный бюллетень или записать на видео (можно на скрытую камеру) процедуру своего волеизъявления. Таким образом, этот пункт хоть и важный, и в эталонной системе голосования должен присутствовать, но при моделировании действующей процедуры его можно опустить.

Таким образом “жизненно важные” свойства концептуальной системы следующие:

1. Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей;
2. Каждый голосует лично, не допускается голосование за кого-то другого;
3. Нельзя узнать как проголосовал кто-то другой;
4. Можно проголосовать только один раз;
5. Нельзя узнать промежуточные результаты.

1.3.2. Полное упрощение

Можно ли еще сильнее упростить модель? Можно, однако тогда потребуется соблюдение определенных условий - тот, кто разворачивает, сопровождает и поддерживает систему, реализующую эту модель, должен быть добропорядочным.

Например, оставим выполняться следующие свойства:

- Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей;
- Каждый голосует лично, не допускается голосование за кого-то другого;
- Можно проголосовать только один раз.

При этом пусть информация о промежуточных результатах и о том, кто как проголосовал хранится в базе в незашифрованном виде, но к ней имеет

доступ только владелец системы. И если он будет честным и не будет разглашать эту информацию, то голосование может быть проведено. Однако это, конечно же, очень частный случай, и в реальном мире нельзя рассчитывать на такие системы.

Дальнейшее упрощение модели ведет к ее полной несостоятельности.

1.4. Модель для реализации

Для реализации возьмем основную модель и выполним над ней три преобразования:

- Добавим пункт о возможности проверки своего голоса избирателем;
- Изменим свойство “Можно проголосовать только один раз” на “Нельзя проголосовать так, чтобы голос был учтен дважды”;
- Пункт “Каждый голосует лично, не допускается голосование за кого-то другого” не будем выделять отдельно, так как он будет учитываться в свойстве “Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей”, потому что будет использоваться инфраструктура публичных ключей (PKI).

Таким образом, результирующий набор свойств, которым удовлетворяет выбранная нами модель:

1. Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей;
2. Нельзя проголосовать так, чтобы голос был учтен дважды;
3. Нельзя узнать как проголосовал кто-то другой;
4. Нельзя узнать промежуточные результаты;
5. Избиратель может проверить, что его голос был учтен.

Глава 2. Архитектура решения

2.1. Описание базового протокола

Предложенный в этой работе подход основан на статье авторов He и Su [18], которые разработали схему электронного голосования без использования технологий распределенных реестров. Она состоит из трех фаз:

1. Регистрация. Тот, кто проводит голосование (удостоверяющий центр, CA), авторизует голосующих (пользователей), которые хотят принять в нем участие. В частности, каждый голосующий генерирует пару ключей (открытый, закрытый). На открытом удостоверяющий центр ставит слепую подпись (Это механизм цифровой подписи с тем дополнительным свойством, что тот, кто предоставляет подпись, не знает, что именно он подписывает. Эта концепция была предложена в [19-20], здесь и далее используется реализация на основе RSA [21]), предварительно убедившись, что пользователь имеет право голосовать.
2. Представление открытого ключа. Голосующий анонимно отправляет подписанный публичный ключ тому, кто будет хранить и подсчитывать бюллетени (счетная комиссия). Она проверяет, что подпись действительно принадлежит CA и в конце сохраняет, а затем публикует все подтвержденные открытые ключи.
3. Голосование и подсчет результатов. Пользователи подписывают бюллетени своими закрытыми ключами и вместе с зашифрованными бюллетенями (шифрование происходит симметричным ключом) анонимно отправляют в счетную комиссию. Та проверяет, что подписи действительны, и публикует все данные, чтобы голосующие могли убедиться, что их бюллетени будут посчитаны. Затем, когда все

зашифрованные бюллетени отправлены, пользователи анонимно посылают симметричные ключи в систему, чтобы можно было расшифровать и подсчитать результаты.

У данной схемы есть ряд недостатков, которые предлагается исправить.

- Бюллетени хранятся в одном месте (у счетной комиссии) и могут быть утеряны (случайно или специально). То есть присутствует единая точка отказа;
- Громоздкий механизм анонимизации (в исходном протоколе предлагается использовать цепочку серверов, проходя через которую сообщение потеряет информацию об отправителе);
- Шифрование бюллетеней с использованием симметричных ключей пользователей. Появляется дополнительная зависимость от голосующих. Необходимо от каждого после сбора бюллетеней получить ключи для их расшифровки. Это чревато тем, что процедура подсчета результатов может затянуться на длительный срок и что часть бюллетеней так и останутся не расшифрованными.

2.2. Модификация базового протокола

Для решения проблемы единой точки отказа предлагается прибегнуть к помощи технологий распределенных реестров (DLT), основные концепции которых следующие:

- Копия реестра (базы данных) находится на каждом узле;
- Нет единой точки отказа;
- Изменить реестр можно, только если большинство узлов согласны с этим;
- Достичь согласия можно, даже если некоторые узлы отключаются во время работы сети.

Блокчейны принято разделять на публичные (стать участником сети может кто угодно) и приватные (до взаимодействия с сетью допускаются только известные участники). К первой группе относятся например Ethereum [22] и Bitcoin [23], ко второй Hyperledger Fabric [24], Exonum [25] и др. Встает вопрос какой блокчейн выбрать, когда их конечных реализаций десятки и даже сотни. Достаточно просто решить, что в случае голосования подойдет приватный блокчейн, так как избиратели должны быть известны и иметь соответствующие права на доступ к системе.

Для дальнейшего выбора необходимо определить требования, предъявляемые в распределенному реестру и на их основе провести сравнение доступных решений. Требования:

1. Наличие средств анонимизации (выполнение этого требования позволит решить проблему наличия стороннего анонимайзера, который требует дополнительной поддержки);
2. Скорость подтверждения транзакций;
3. Открытый исходный код;

4. ГОСТ криптография (необходима для функционирования системы в государственном секторе);
5. Возможность реализации мобильного клиента. Это важно для хранения приватного ключа на устройстве конечного пользователя.

	Анонимизация	Скорость	Открытый код	ГОСТ	Мобильный клиент
Hyperledger fabric [24]	+	до 20000	+	+	+
Quorum [26]	+	до 2500	+	-	+
Waves [27]	+	до 1000	+	+	+
NEO [28]	+	до 10000	+	-	+
Exonum [25]	-	до 7000	+	-	+

Табл.1 Сравнение блокчейн-платформ

Стоит отметить, что показателя в 20000 транзакций в секунду Hyperledger Fabric достигает при условии существенной модификации непосредственно узлов и архитектуры платформы (при настройках по умолчанию разработчиками заявляется скорость до 3000). Но даже с учетом этого факта, данный блокчейн является выигрышным, потому что удовлетворяет всем остальным требуемым свойствам.

Таким образом, распределенным реестром, на котором было принято решение реализовывать систему, стал Hyperledger Fabric. Это приватный блокчейн, разрабатываемый и поддерживаемый компанией IBM. Основное его отличие от других платформ заключается в порядке обработки транзакций:

1. Подтверждение. Транзакции проверяются на возможность исполнения, затем симулируется их выполнение. При этом состояние реестра не меняется;

2. Упорядочивание. Узлы, отвечающие за расположение транзакций в блоке, упорядочивают их;
3. Проверка и запись. Упорядоченные транзакции отправляются на все узлы, которые проверяют достоверность подписей на транзакциях и обновляют свои копии реестра.

В соответствии с таким разделением принятия новой транзакции в реестр существует три типа узлов: Endorsing Peer, которые выполняют (симулируют) транзакции (на них исполняется логика), Ordering Nodes, производящие упорядочивание, Committing Peer, поддерживающие копию реестра и обновляющие его (Endorsing Peer также является и Committing Peer).

Исходный код данной платформы распространяется под лицензией Apache 2.0, которая предусматривает возможность свободно изменять и распространять данный продукт.

Возможность ГОСТ шифрования, сертифицированного ФСБ (что необходимо для использования в государственном секторе), обеспечивается модулем интеграции КриптоПро CSP, разработанным специалистами из одноименной компании [29].

Наличие инструментов для реализации мобильных клиентов, является одним из важных требований, которые предъявляются блокчейн-платформе. Это происходит из-за того, что необходимо минимизировать возможность попадания приватных ключей пользователя к третьим лицам. Это можно сделать, если ключи будут генерироваться и храниться непосредственно на устройстве пользователя. И смартфон с мобильным приложением является достаточно хорошим решением. Взаимодействие же через веб-браузер сопряжено с рисками компрометации пользовательских сертификатов. Это может произойти, либо если ключи хранятся на серверах (приходится полагаться на их надежность), либо при использовании локального

хранилища в браузере (приходится использовать сторонние расширения, например Metamask [30], которые могут быть подвержены фишинговым атакам).

Hyperledger Fabric предоставляет SDK для разработки клиентских приложений на языках Java, Golang и др. Java дает возможность взаимодействовать с реестром через Android. А инструмент gomobile [31] позволяет либо писать нативные приложения под iOS, Android, либо делать вызовы из кода на Objective-C и Java.

Помимо сказанного выше платформа Hyperledger Fabric имеет встроенный механизм анонимизации Identity Mixer (Idemix), который будет использован для решения проблемы сокрытия выбора голосующего. Данная технология разработана компанией IBM и основана на криптографических техниках, доказательства математической корректности которых были представлены на передовых научных конференциях и тщательно проверены крипто-сообществом. Практическая реализация основана на схеме, с которой можно подробнее ознакомиться в [32-34]. Здесь мы приведем лишь краткое описание данной технологии.

Предположим, что есть есть три участника: пользователь, издатель (issuer, орган, выдающий сертификаты и подтверждающий личность пользователя) и проверяющий (verifier), который хочет убедиться в том, что пользователю действительно присущи некоторые атрибуты, которые необходимы для выполнения определенного действия. В такой схеме основная проблема, которую решает Idemix, это предоставление проверяющему не всей информации о пользователе, а только определенных атрибутов или даже просто доказательств того, что запрашиваемый атрибут попадает в определенный диапазон. К тому же есть возможность обеспечить отсутствие связи двух разных запросов от одного и того же пользователя с самим этим пользователем. Таким образом, Idemix позволяет обеспечить

анонимность и приватность при взаимодействии конечного избирателя с реестром.

Осталась проблема с шифрованием бюллетеней. Как было сказано раньше, когда каждый из голосующих шифрует свой голос своим собственным ключом, есть немалая вероятность, что часть бюллетеней не будет расшифрована и окажется неучтенной в конечном результате. Для того чтобы минимизировать этот риск, предлагается использовать один общий ключ шифрования. Но здесь возникает проблема необходимости доверия тому, кто хранит приватную часть этого ключа. Теоретически он может во время голосования расшифровать текущие бюллетени и узнать промежуточные результаты, а это противоречит одному из свойств модели, которая реализуется. Избежать этого можно используя схему разделения секрета. Этот подход позволяет распределить некоторую информацию (в нашем случае это приватная часть ключа шифрования) между несколькими участниками так, чтобы восстановить ее можно было, только если все (или большинство) участников предоставят свои отдельные части ключа. Многие схемы являются частным случаем более общего теоретического подхода [35]. Таким образом, предлагается разделить приватную часть ключа для шифрования бюллетеней между несколькими доверенными наблюдателями. В случае выборов это могут быть независимые общественные организации, а также наблюдатели от разных партий.

2.3. Описание архитектуры

В этом разделе описывается архитектура решения, положенная на платформу Hyperledger Fabric. Будем рассматривать сценарий, в котором наравне с возможностью электронного голосования, часть избирателей будет голосовать традиционно бумажными бюллетенями.

Рассмотрим сначала высокоуровневое описание схемы голосования. Некий организатор хочет провести выборы, в которых предусмотрена возможность электронного голосования. Объявляется дедлайн регистрации, до этого момента любой избиратель, которому позволено голосовать, может изъявить желание сделать это электронно. Он заходит в систему с помощью учетных данных, полученных от организатора выборов (это могут быть например сертификаты x.509) и делает отметку о том, что будет голосовать через систему (впоследствии он может изменить свое решение, если время регистрации не закончилось). После дедлайна списки тех, кто решил голосовать электронно, передаются в структуры, ответственные за проведение выборов традиционным образом, чтобы исключить возможность избирателя проголосовать и очно, и через систему. Когда начинается непосредственно промежуток волеизъявления, участники, голосующие электронно, заходят в систему и делают свой выбор. Пока голосование не закончилось, присутствует возможность изменить решение. По окончании данной процедуры публикуются результаты, достоверность которых каждый может проверить с помощью системы.

Рассмотрим участников процесса:

1. **Org** - некий центральный орган, проводящий голосование (является организацией в терминах Hyperledger Fabric, имеет один или несколько физических узлов);
2. **Dep** - подразделение (территориальная избирательная комиссия), которое отвечает за проведение голосования в своем округе (например в определенное время начинает процесс голосования, это может быть полезно там, где есть несколько часовых поясов). Является либо отдельной организацией, либо подразделением центрального органа, имеет один или несколько физических узлов;

3. **CC** - умный контракт (chaincode в терминах Hyperledger Fabric), который отвечает за логику проведения выборов;
4. **V** - избиратель (пользователь системы), который является членом одного из подразделений;
5. **Ins** - наблюдатель (организация в терминах Hyperledger Fabric), имеет один или несколько узлов, следит за соблюдением правил проведения выборов.

Фазы:

1. Конфигурация сети;
2. Конфигурация голосования;
3. Регистрация пользователей;
4. Голосование;
5. Подсчет результатов.

2.3.1. Конфигурация сети

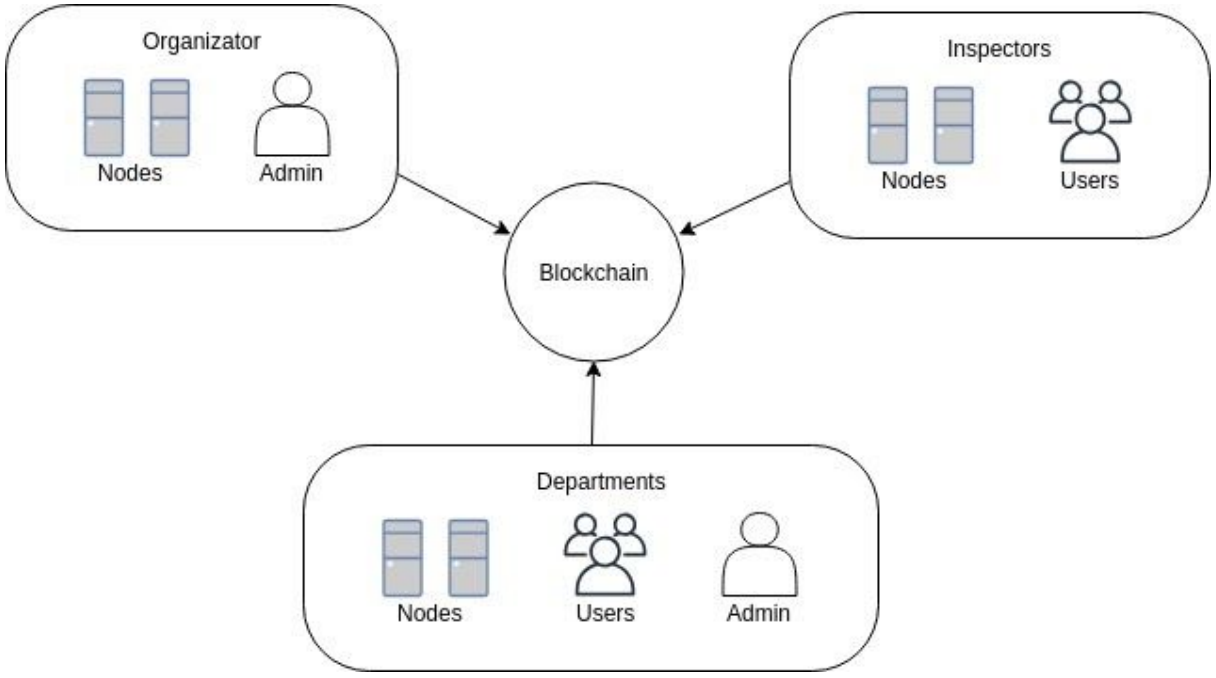

Рис. 1. Высокоуровневая схема решения.

Во время этой фазы определяются организации, количество имеющихся у них узлов, права (чтение, запись). Поднимается сеть,

добавляются необходимые узлы, на которых расположены CA для каждой из организаций, загружается логика (chaincode), определяются узлы типа ordering, которые будут принимать участие в достижении консенсуса над включением транзакций в реестр (Рис. 1).

2.3.2. Конфигурация голосования

Во время этой фазы (Рис. 2) администратор **Org** формирует список вопросов, определяет список **Dep**, которые могут принимать участие в этом голосовании. Эти данные загружаются в реестр. Затем каждый из администраторов **Dep** формирует дополнительные данные (начало/конец голосования, начало/конец регистрации, список голосующих), необходимые

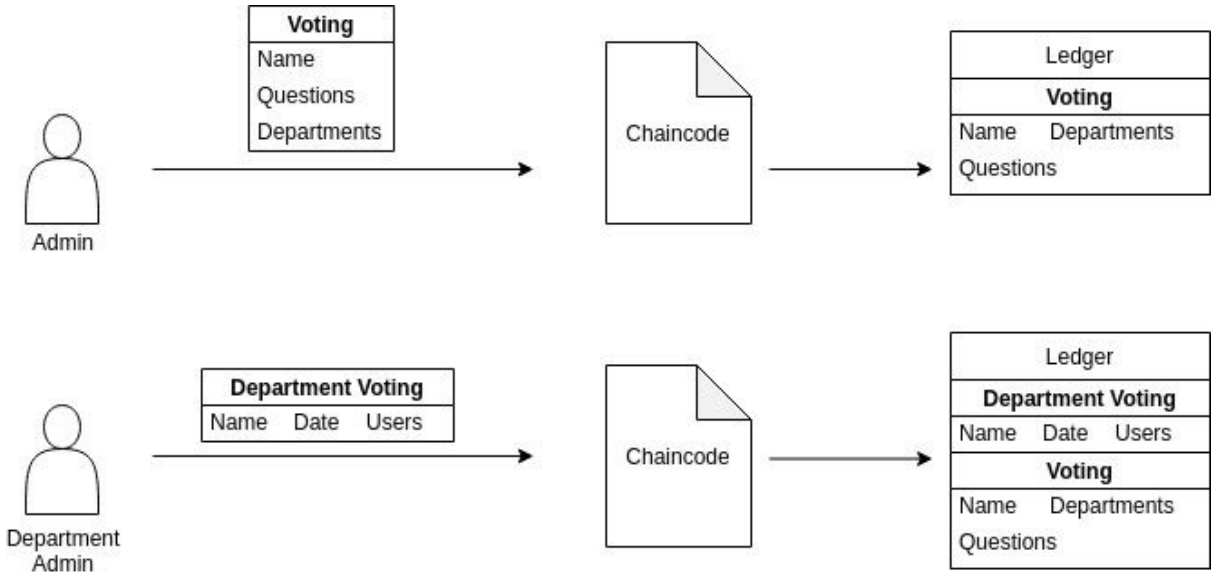

Рис. 2. Схема конфигурации голосования.

для проведения процедуры голосования среди избирателей, которые принадлежат к этому подразделению. Далее будем называть эти данные локальными выборами. Они тоже загружаются в реестр. Также на данном этапе генерируются пары ключей для каждого подразделения (E_i – публичный, D_i – приватный), i от 1 до n, где n – количество подразделений. Публичный ключи записывается в реестр, а приватный сохраняется в private data collection (специальный механизм в hyperledger fabric, который позволяет ограничить доступ к данным). После того как

данные загружены, для каждого пользователя обновляется список доступных для него голосований.

2.3.3. Регистрация пользователей

Эта фаза (Рис. 3) необходима для нескольких целей: во-первых, чтобы сохранить возможность избирателей голосовать с помощью бумажных бюллетеней (если пользователь не прошел регистрацию, то он сможет голосовать только традиционно); во-вторых, она позволит обеспечить анонимность выбора, в то же время сохранив свойство доступности голосования только авторизованным пользователям.

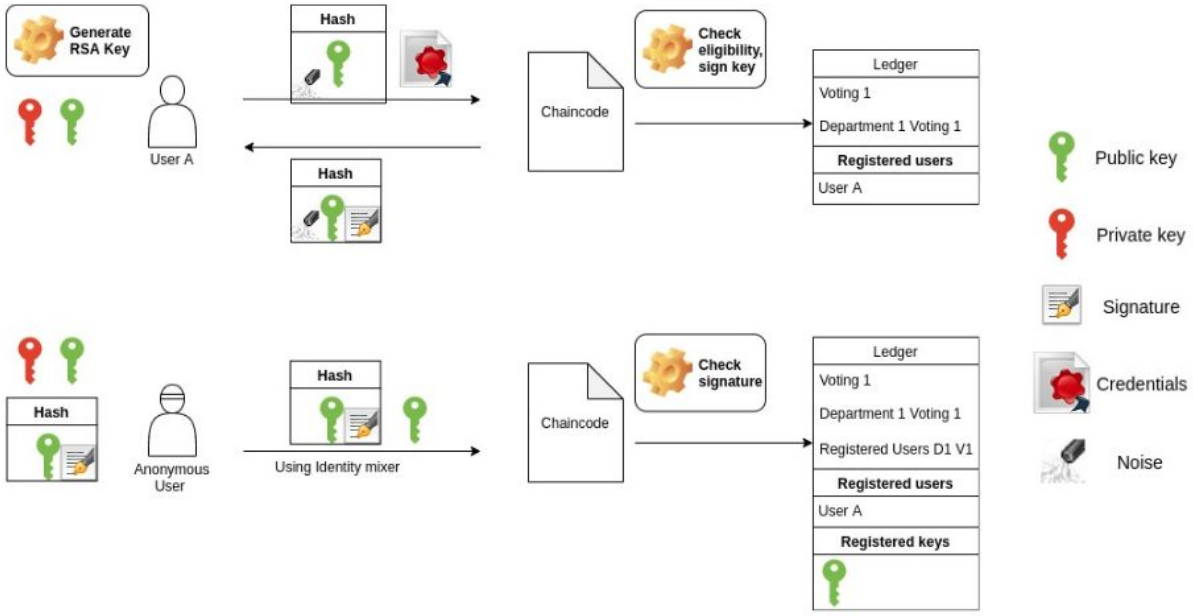

Рис. 3. Схема регистрации пользователей в голосовании.

Во время подэтапа получения слепой подписи каждый пользователь генерирует пару ключей E_v (публичный), D_v (приватный) и случайное число R. Затем вычисляет $E_i(R) * h(E_v)$, где $E_i(R)$ – случайное число, зашифрованное с помощью публичного ключа E_i подразделения, которому принадлежит пользователь, а $h(E_v)$ - хэш-функция над публичным ключом

избирателя. Это произведение вместе с данными о тех выборах, в которых хочет зарегистрироваться избиратель, отправляется в **CC**, отвечающий за логику регистрации. Там происходят проверки того, что текущему пользователю действительно позволено участие в этих выборах на основе данных (имя пользователя, подразделение) от CA. Если все верно, то $E_i(R) * h(E_v)$ подписывается приватным ключом подразделения D_i. Полученные данные $D_i(E_i(R) * h(E_v)) = R * D_i(h(E_v))$ (равенство верно, так как используются ключи по типу RSA) отправляются обратно пользователю, при этом в реестр записывается информация о том, что текущий пользователь получил слепую подпись. В конце фазы регистрации списки таких избирателей можно использовать для того, чтобы забрать у них возможность бумажного голосования (если такое предусмотрено). Избиратель проверяет достоверность полученных данных, удаляя R из этих данных ($R * D_i(h(E_v)) * R^{-1}$, R^{-1} – обратное к R по модулю публичного ключа департамента) и проверяя, что хэш подписан действительно его подразделением, сравнив $h(E_v)$ и $E_i(D_i(h(E_v)))$.

Далее следует подэтап регистрации публичного ключа, связь которого с конкретной личностью знает только сам избиратель. В этот момент пользователь имеет подписанный публичный ключ $D_i(h(E_v))$, который он вместе с E_v анонимно при помощи Identity mixer (подробнее в следующем разделе) отправляет в **CC,** где проверяется $h(E_v)$ и $E_i(D_i(h(E_v)))$, и в случае успеха публичный ключ E_v записывается в реестр (во время фазы голосования запросы от избирателей будут анонимными (используется технология Idemix), а с помощью данных опубликованных ключей система сможет убедиться в том, что пользователю действительно позволено голосовать.

Для того чтобы пользователь снова получил возможность голосовать бумажно, ему необходимо выписаться из списка тех, кто получил слепую подпись на первом подэтапе регистрации. Эта фаза происходит не анонимно. Избиратель отправляет в **CC** запрос на удаление ключа регистрации E_v из списка. **CC** не удаляет ключ E_v совсем, а помечает его как “отозван” (это нужно для того, чтобы если пользователь передумает и захочет снова зарегистрироваться он регистрировал новый ключ). Помимо этого происходит удаление его из списка подписанных пользователей (опять же с пометкой “отозван”). Если избиратель снова захочет принять участие в электронных выборах он просто переходит на первый подэтап и формирует новую пару ключей.

2.3.4. Голосование

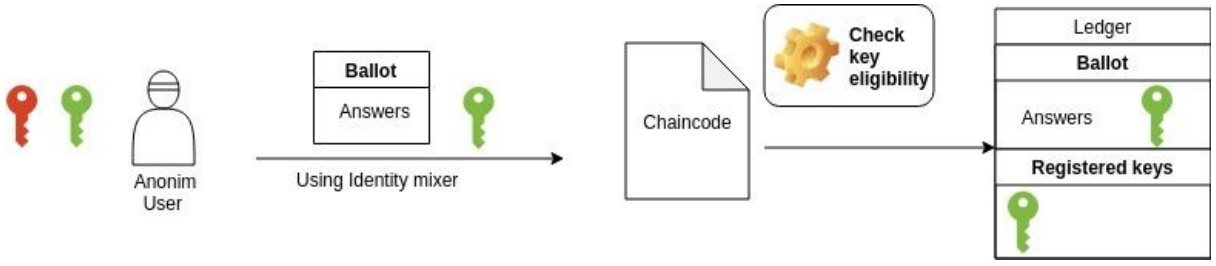

Рис. 4. Схема голосования пользователей.

Во время этой фазы пользователи, которые зарегистрировали свои публичные ключи для данного голосования, могут отправлять зашифрованные бюллетени в **CC** (Рис. 4). Пока время голосования не закончилось, избиратель может изменить свой выбор. Учитываться будет только последний по времени голос. Как было сказано ранее, для шифрования бюллетеня клиенты используют общий публичный ключ, секретная часть которого разделена между независимыми наблюдателями.

2.3.5. Подсчет результатов

После завершения голосования участники, имеющие части приватного ключа, загружают их в **CC**. Таким образом, появляется возможность

расшифровать бюллетени и получить результаты, которые публикуются в реестр.

2.3.6. Присутствие инспекторов

Для увеличения доверия к процедуре проведения выборов в систему введены инспекторы, которые предполагаются независимыми и неподкупными. Они имеют физические узлы, на которых хранится копия реестра. Однако так как эти узлы являются узлами третьего типа (committing peer), то наблюдатели не способны повлиять на решение о принятии или отклонении транзакци, но могут обнаруживать случаи недобросовестного поведения других участников.

2.4. Анализ безопасности

В этом разделе будет описано почему предлагаемое решение соответствует требованиям, которые были заявлены.

Принимать участие в голосовании могут только те избиратели, которые включены в список избирателей. Это требование обеспечивается с помощью сертификатов, которые есть у каждого пользователя для взаимодействия с сетью (например x509, стоит отметить, что использование Hyperledger Fabric позволяет не создавать новые сертификаты, а использовать принятые в конкретной организации), и механизма регистрации, описанного выше. Так как в рассматриваемом сценарии предусмотрена возможность голосовать либо электронно, либо бумажно, то возникает проблема точного определения пользователя в одну из этих двух групп. Это необходимо для того, чтобы исключить возможность голосующего принять участие в выборах и электронно, и бумажно. Это достигается за счет формирования списка тех, кого необходимо вычеркнуть из перечня лиц, допущенных к бумажному голосованию, на первом этапе регистрации пользователя. Однако необходимо указать один недостаток такого подхода. Так как регистрация в

электронном голосовании состоит из двух шагов (и принять участие в выборах можно только, если ты завершил вторую фазу), а из перечня лиц, допущенных к бумажному голосованию, исключаются списки, полученные после первой фазы регистрации, то возможна ситуация, когда пользователь не завершит вторую фазу и, следовательно, не сможет принять участие в голосовании ни через систему, ни традиционным способом, хотя имеет на это полное право.

Нельзя проголосовать так, чтобы голос был учтен дважды. Избиратель может проголосовать только если его идентификатор (публичный ключ) присутствует в списке зарегистрированных ключей для данных выборов. Пользователь не может зарегистрировать два ключа, так как на каждом ключе должна стоять подпись соответствующих выборов, но при получении подписи на первом ключе, пользователь вносится в список и, при попытке получить подпись на другом ключе, он получит отказ.

Нельзя узнать как проголосовал кто-то другой. Во время голосования права пользователя определяются на основе его анонимного идентификатора (только избиратель знает, что конкретный id принадлежит ему). И так как все запросы посылаются через Idemix, то в системе нет связи id, а следовательно и голоса, с реальным избирателем.

Нельзя узнать промежуточные результаты. Это свойство выполняется, так как все бюллетени шифруются и не могут быть раскрыты до конца голосования. Теоретически, если все независимые наблюдатели (между которыми распределен приватный ключ шифрования) вступят в сговор с организатором, промежуточные результаты могут стать известны. Но вероятность этого мала.

Избиратель может проверить, что его голос был учтен. Так как все голоса записаны в реестр с привязкой к id, то пользователь может сам

пересчитать результаты, учитывая свой голос (ему известен его id), тем самым убедившись, что его выбор был учтен.

Глава 3. Реализация и тесты

Данная система была разработана и прошла государственную регистрацию, а также ее компоненты были оформлены как РИД (результат интеллектуальной деятельности), ссылки на которые можно найти в приложении.

Было проведено вычисление пропускной способности Hyperledger Fabric (в наиболее общих конфигурациях) для сценариев голосования, в рамках которого были получены достаточно скромные результаты. Скорость была от 1000 до 2000 транзакций в секунду, хотя статья от самих разработчиков [36] утверждает, что их решение может обеспечить пропускную способность в пределах от 1000 до 3500 tps. Видимо, это происходит из-за того, что их тестирование проводилось на очень простых приложениях, которые не требовали существенных ресурсов. При этом некоторые исследователи [37] показывают, что при серьезной модификации определенных компонентов можно достичь скорости в 20000 транзакций в секунду. Однако в то же время исследователи из Сбербанка [38] показали, что в некоторых случаях скорость будет около 1000 tps при использовании встроенной базы данных LevelDB, и около 300 tps, при использовании CouchDB.

Как уже было сказано, тесты, которые проводились в рамках данной работы, показали промежуточные результаты между этими крайними значениями. Так были протестированы несколько различных конфигураций сети:

- Solo (одна организация, один узел, один orderer, всего 5 docker-контейнеров)

- Kafka (две организации, в каждой по 2 узла, два orderer, 3 zookeeper, 4 kafka контейнера, всего 20 docker-контейнеров)

Каждая показала схожие характеристики - порядка 2000 tps. При большей нагрузке некоторые транзакции перестали проходить из-за таймаута (который был выставлен в 60 секунд). Клиенты для взаимодействия с реестром были написаны с использованием следующих SDK:

1. Java SD;
2. Node SDK;
3. Python SDK.

Ниже представлены графики среднего времени выполнения сценария голосования для одного пользователя в мс (При разном количестве единовременно активных пользователей. От 100 до 1300 с шагом 200. При большем количестве пользователей система переставала справляться с нагрузкой и либо падала, либо часть пользователей не могла получить ответ от системы).

Сценарий включал в себя следующие стадии:

1. Регистрация пользователя в системе HLF (помечено как reg на графиках);
2. Получение доступных голосований (у каждого пользователя было доступно два голосования, одно из которых уже закончилось);
3. Получение результатов закончившегося голосования;
4. Регистрация в идущем голосовании;
5. Голосование.

При этом в реестр сохранялись следующие данные:

- Информация о голосовании;
- Информация о голосующих;
- Бюллетени;
- Результаты.

Тестирование проводилось на различных конфигурациях сети (solo, kafka). Также изучалось влияние регистрации (если у пользователя еще нет ключей для взаимодействия с реестром, и ему необходимо обратиться к CA, чтобы их получить).

Пример названия конфигурации: “solo 2, 10, 99 (reg)” означает, что поднималась solo сеть со следующими параметрами формирования блоков:

- Batch Timeout: Время, которое необходимо для создания блока - 2s;
- Max Message Count: Максимальное количество сообщений в блоке - 10 msg;
- Absolute Max Bytes: Максимальное количество байт, разрешенное для сериализации сообщения в блоке - 99 MB;
- Preferred Max Bytes: Желаемое максимальное количество байт, разрешенное для сериализации сообщения в блоке - 512 kb;

При этом пользователям приходилось сначала регистрироваться (пометка reg в названии конфигурации).

Если в названии конфигурации присутствует четвертый параметр, это значит, что параметр формирования блока "PreferredMaxBytes" равен 2 MB, в противном случае он равен 512 kb.

График 1 показывает сравнение времени работы в одних и тех же конфигурациях в зависимости от необходимости изначально регистрироваться в системе. Видно, что при необходимости регистрации время выполнения сценария увеличивается в 1.5-2 раза. Это связано с тем, что при регистрации пользователь получает новый X.509 сертификат, формирование которого является дорогостоящей криптографической операцией.

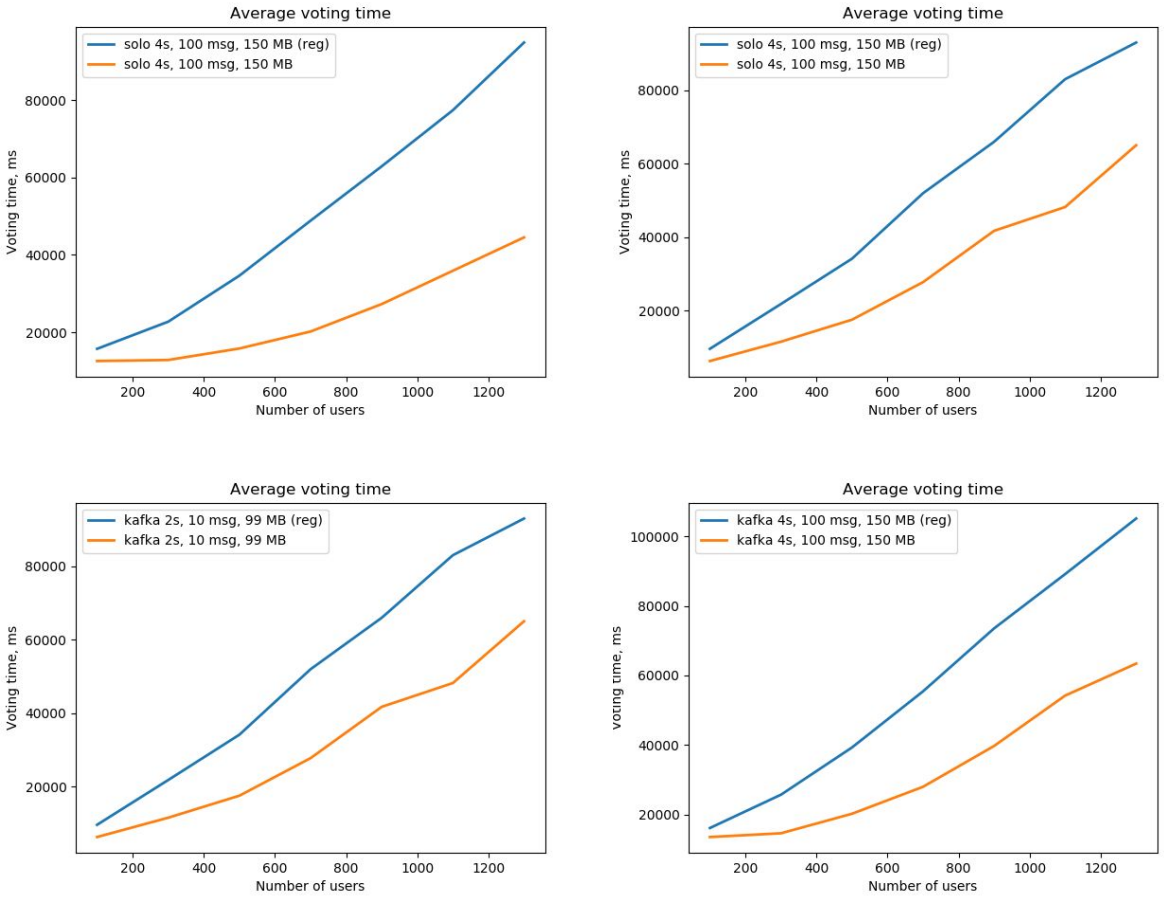

График 1. Сравнение времени выполнения сценария в зависимости от необходимости регистрации в системе.

График 2 представляет сравнение времени работы разных типов консенсуса (solo и kafka) в одних и тех же конфигурациях. Время выполнения сценария в случае solo ожидаемо оказалось меньше из-за того, что фактически есть только один узел, который упорядочивает транзакции, что позволяет не тратить ресурсы на консенсус между узлами как это происходит в kafka.

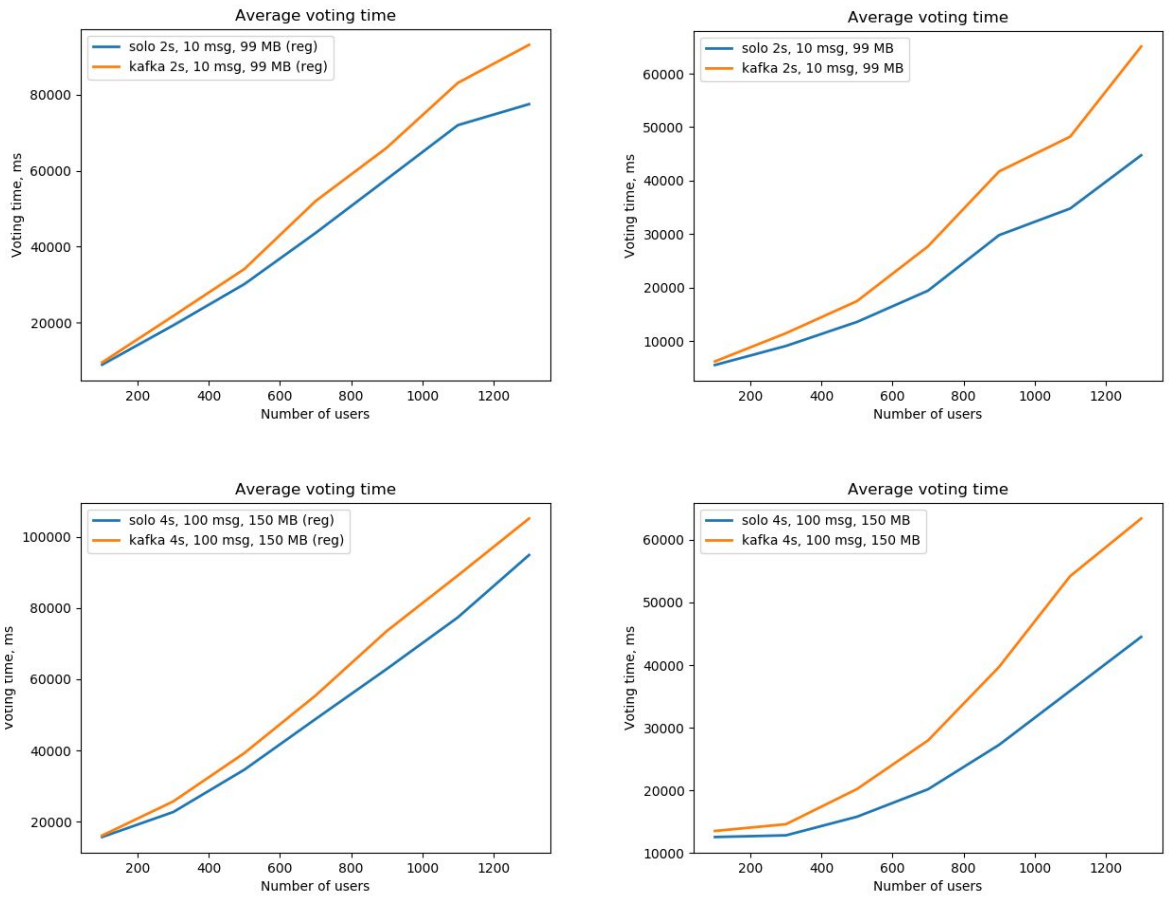

График 2. Сравнение времени работы при разных типах консенсуса.

График 3 показывает время работы при одном и том же типе консенсуса, но с разными параметрами формирования блока. Видно, что увеличение размера блока в случае, когда регистрация требовалась, не дало прироста производительности. Это связано с тем, что сама регистрация занимает по времени сопоставимое количество секунд с выполнением всего остального сценария. В случае же ее отсутствия в консенсусе kafka мы видим прирост, особенно заметный после 900 одновременно активных пользователей.

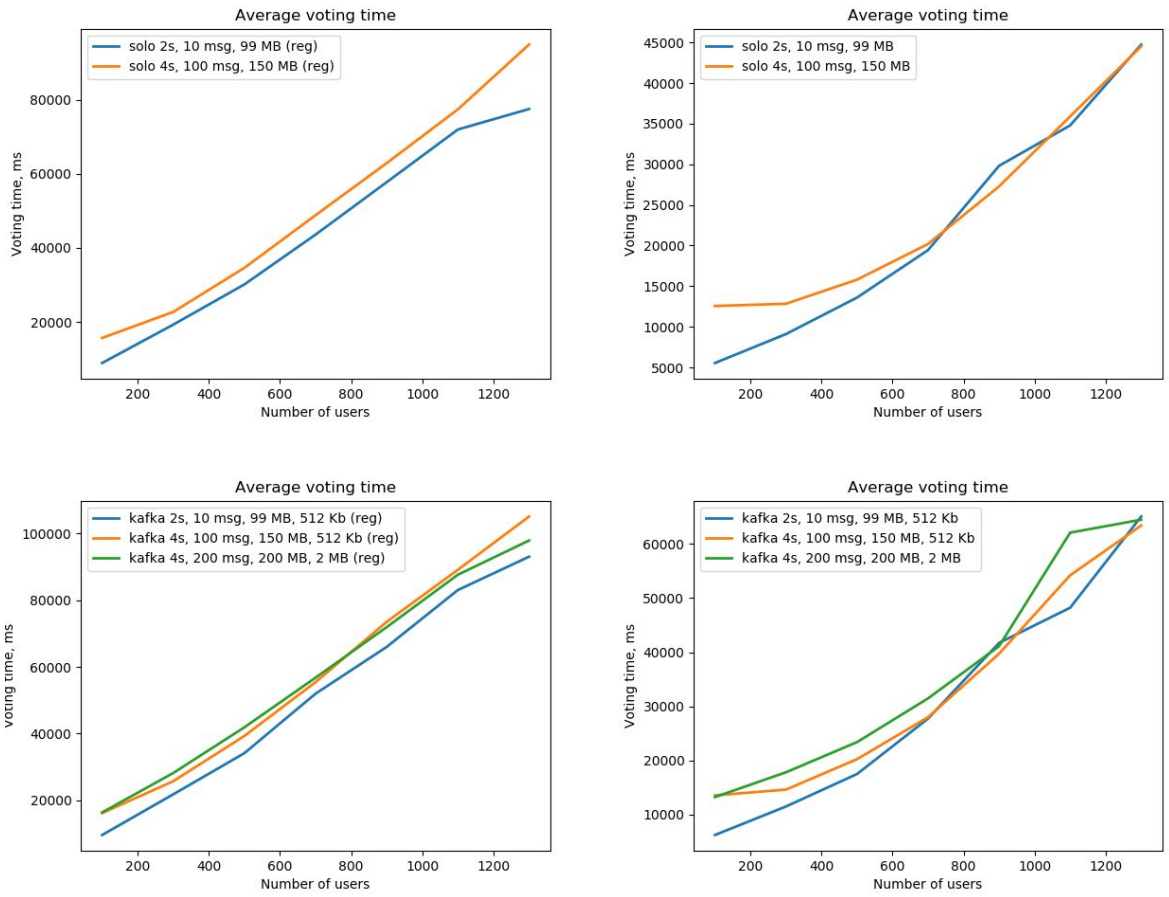

График 3. Сравнение времени выполнения при различных параметрах формирования блока.

Выводы

Несмотря на то, что все заявленные требования были удовлетворены и система обеспечивает должный уровень распределенности и анонимности, цена, которой это достигается, достаточно высока. Скорость выполнения транзакций оказывается недостаточной для проведения голосования на миллионы пользователей, что соответствует, например, реальным выборам. К сожалению, невысокая скорость обработки транзакций - это основная проблема всех современных блокчейн-платформ. Однако следует отметить, что индустрия развивается в этом направлении и предлагает подходы к преодолению существующих ограничений. Один из таких подходов (наиболее популярный и перспективный) это шардинг [39].

Дальнейшее улучшение разработанной системы может быть связано с удовлетворением дополнительных требований в расширенной модели из главы 1. В частности можно рассмотреть добавление кольцевых связанных подписей, чтобы обеспечить отсутствие у пользователя доказательств того, как именно он проголосовал.

Заключение

В работе были обозначены несколько моделей электронного голосования, которые основаны на свойствах, предъявляемых к процедуре голосования российским законодательством. Для реализации одной из этих моделей была модифицирована схема электронного голосования. В нее были внесены изменения, которые помогли решить следующие проблемы:

1. Единая точка отказа и недоверие к данным;
2. Громоздкий механизм анонимизации;
3. Зависимость расшифровки бюллетеней от каждого голосующего.

Так как решение первой проблемы потребовало использование технологий распределенных реестров, то была проведена исследовательская работа по сравнению и выбору наиболее подходящей платформы, которой оказалась Hyperledger Fabric.

Тестирование разработанного решения на производительность показало, что одной из главных проблем технологий распределенных реестров остается скорость выполнения транзакций, которая на данный момент может достигать в лучшем случае отметки в пару десятков тысяч.

В результате можно заключить, что система, представленная в данной работе, может быть использована для относительно небольших голосований (несколько тысяч участников) без дополнительных модификаций. При этом будут выполняться основные требования, предъявляемые к процедуре голосования.

Список литературы

1. Madise, U., Martens, T. E-voting in estonia 2005. the first practice of country-wide binding internet voting in the world // Electronic Voting 2006: 2nd International Workshop. 2006. N. 86 C. 15-26.
2. Brightwell, I., Cucurull, J., Galindo, D., Guasch, S. An overview of the ivote 2015 voting system // The Fifth International Conference on Voting and Identity. 2015.
3. Springall D., Finkenauer T., Durumeric Z., Kitcat J., Hursti H., MacAlpine M., Halderman J.A. Security analysis of the estonian internet voting system // Conference on Computer and Communications Security. 2014. C. 703-715.
4. Halderman J.A., Teague V. The new south wales ivote system: Security failures and verification flaws in a live online election // E-Voting and Identity - 5th International Conference. 2015. N 9269. C. 35-53.
5. Volkhausen T. Paillier cryptosystem: A mathematical introduction // Seminar Public-Key Kryptographie. 2006.
6. Hardwick S.F., Apostolos G., Naeem A.R., Konstantinos M. E-Voting with Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy. 2018.
7. Hjálmarsson F., Hreiðarsson G.K. Blockchain-Based E-Voting System // IEEE 11th International Conference on Cloud Computing. 2018. C. 983-986.
8. Yavuz E., Koç A.K., Çabuk U.C., Dalkılıç G. Towards secure e-voting using ethereum blockchain, 2018, 6th International Symposium on Digital Forensic and Security. 2018.C. 1-7.
9. Liu Y., Wang Q. An E-voting Protocol Based on Blockchain // IACR Cryptol. ePrint Arch. 2017.

10. Yu B., Liu J., Sakzad A., Nepal S., Steinfeld R., Rimba P., Au M.H. Platform-independent Secure Blockchain-Based Voting System // International Conference on Information Security. 2018. C. 369-386.
11. Au M.H., Chow S.S., Susilo W., Tsang P.P. Short linkable ring signatures revisited // European Public Key Infrastructure Workshop. 2006. C. 101-115.
12. Menon A., Bhagat V. Blockchain based e-voting system // Indian Journal of Applied Research. 2020. N 10.
13. Yi H. Securing e-voting based on blockchain in P2P network // J Wireless Com Network. 2019, N 137.
14. Churyumov A. Byteball: A Decentralized System for Storage and Transfer of Value [Электронный ресурс]: URL: https://byteball.org/Byteball.pdf (дата обращения: 27.05.2020).
15. Shiyao G., Dong Z., Rui G., Chunming J., Chencheng H. An Anti-Quantum E-Voting Protocol in Blockchain with Audit Function // IEEE Access. 2019. N 7. C. 115304-115316.
16. Khan K. M., Arshad J., Khan M. M., Investigating performance constraints for blockchain based secure e-voting system // Future Generation Computer Systems, N 105. 2020, C. 13-26.
17. Liu J.K., Wong D.S. Linkable ring signatures: Security models and new schemes // International Conference on Computational Science and Its Applications. 2005 C. 614-623.
18. He Q., Su Z. A New Practical Secure e-Voting Scheme // 14th International Information Security Conference. 1998.
19. Chaum D. Blind Signatures for Untraceable Payments // Advances in Cryptology. 1983. C. 199-203.
20. Chaum D. Security without identification: transaction systems to make big brother obsolete // Communications of the ACM. 1985. N 28. C. 1030-1044.

21. Rivest R., Shamir A., Adleman L. A method for obtaining digital signatures and public-key cryptosystems // Communications of the ACM. 1978. N 21. C. 120-126.
22. Проект Ethereum [Электронный ресурс]: URL: https://ethereum.org/ru/ (дата обращения: 27.05.2020).
23. Проект Bitcoin [Электронный ресурс]: URL: https://bitcoin.org/ru/how-it-works (дата обращения: 27.05.2020).
24. Проект Hyperledger Fabric [Электронный ресурс]: URL: https://www.hyperledger.org/use/fabric (дата обращения: 27.05.2020).
25. Проект Exonum [Электронный ресурс]: URL: https://exonum.com/index (дата обращения: 27.05.2020).
26. Проект Quorum [Электронный ресурс]: URL: https://www.goquorum.com/ (дата обращения: 27.05.2020).
27. Проект Wawes [Электронный ресурс]: URL: https://wavesprotocol.org/ (дата обращения: 27.05.2020).
28. Проект NEO [Электронный ресурс]: URL: https://neo.org/ (дата обращения: 27.05.2020).
29. Модуль интеграции ГОСТ криптографии CryptoPro CSP для блокчейн-платформы Hyperledger Fabric [Электронный ресурс]: URL: http://crypto-pro.ru/news/2019/03/modul-integratsii-cryptopro-csp-s-hyperledger-fabric (дата обращения: 27.05.2020).
30. Проект Metamask [Электронный ресурс]: URL: https://metamask.io/ (дата обращения: 27.05.2020).
31. Инструмент gomobile для разработки [Электронный ресурс]: URL: https://pkg.go.dev/golang.org/x/mobile/cmd/gomobile?tab=doc (дата обращения: 27.05.2020).

32. Camenisch J., Lysyanskaya A. Signature Schemes and Anonymous Credentials from Bilinear Maps // Advances in Cryptology. 2004. N 3152. C. 56-72.
33. Au M.H., Susilo W., Mu Y. Constant-Size Dynamic *k*-TAA // *IEEE Systems Journal*. 2006. N 7. C. 249-261.
34. Camenisch J., Drijvers M., Lehmann A. Anonymous Attestation Using the Strong Diffie Hellman Assumption Revisited // IACR Cryptology ePrint Archive. 2016. N 663.
35. Kothari S.C. Generalized Linear Threshold Scheme // Advances in Cryptology. 1984. N 196. C. 231-241.
36. Androulaki E., Barger A., Bortnikov V., Cachin C., Christidis K., Caro A., Enyeart D., Ferris C., Laventman G., Manevich Y., Muralidharan S., Murthy C., Nguyen B., Sethi M., Singh G., Smith K., Sorniotti A., Stathakopoulou C., Vukolic M., Yellick J., Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains // EuroSys '18: Proceedings of the Thirteenth EuroSys Conference. 2018. C. 1-15.
37. Gorenflo C., Lee S., Golab L., Keshav S. FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second // IEEE International Conference on Blockchain and Cryptocurrency. 2019. C. 455-463.
38. Ссылка на доклад разработчиков Сбербанка о тестирование платформы Hyperledger Fabric [Электронный ресурс]: URL: https://wiki.hyperledger.org/display/RU/Stress+test+HLF+network+with+basic+Dapp (дата обращения: 28.05.2020).
39. Информация о шардинге [Электронный ресурс]: URL: https://ru.bitcoinwiki.org/wiki/%D0%A8%D0%B0%D1%80%D0%B4%D0%B8%D0%BD%D0%B3 (дата обращения: 28.05.2020).

Приложение

Ссылки на РИДы:

1. Информационная система администратора платформы для проведения электронного голосования на базе технологий распределенных реестров (Программа для ЭВМ), 18.06.2019 №219.017.83bc https://edrid.ru/rid/219.017.83bc.html
2. Информационная система клиента платформы для проведения электронного голосования на базе технологий распределенных реестров (Программа для ЭВМ), 18.06.2019 №219.017.83bd https://edrid.ru/rid/219.017.83bd.html
3. Смарт-контракты распределенного реестра для проведения доверенного электронного голосования с использованием платформы Hyperledger Fabric (Программа для ЭВМ), 18.06.2019 №219.017.83be https://edrid.ru/rid/219.017.83be.html
4. Библиотека для взаимодействия с распределенным реестром Hyperledger Fabric в платформе для проведения электронного голосования (Программа для ЭВМ), 18.06.2019 №219.017.83bf https://edrid.ru/rid/219.017.83bf.html
5. Платформа для проведения электронного голосования на базе технологий распределенных реестров (Программа для ЭВМ), 18.06.2019 №219.017.83c0 https://edrid.ru/rid/219.017.83c0.html

Ссылка на госрегистрацию:

1. Государственная регистрации программы для ЭВМ: “Платформа для электронного голосования на базе технологий распределенных реестров” https://fips.ru/registers-doc-view/fips_servlet?DB=EVM&DocNumber=2020612936&TypeFile=html